IFW
AR/6

PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of

Richard H. MANDEL III

Appln. No.: 09/545,592

Confirmation No.: 9939

Filed: April 07, 2000

For: CROSS-PLATFORM SUBSELECT METADATA EXTRACTION

Docket No: A8496 / ST9-99-179

Group Art Unit: 2172

Examiner: Hung Q. Pham

SUBMISSION OF APPEAL BRIEF

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Submitted herewith please find an Appeal Brief. A check for the statutory fee of $500.00 is attached. The USPTO is directed and authorized to charge all required fees, except for the Issue Fee and the Publication Fee, to Deposit Account No. 19-4880. Please also credit any overpayments to said Deposit Account. A duplicate copy of this paper is attached.

Respectfully submitted,

for Timothy P. Cremen
Registration No. 50,855

THE REMARK WILLIAMS #47,177

SUGHRUE MION, PLLC
Telephone: (202) 293-7060
Facsimile: (202) 293-7860

WASHINGTON OFFICE
23373
CUSTOMER NUMBER

Date: October 21, 2005

PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of

Richard H. MANDEL III

Appln. No.: 09/545,592

Confirmation No.: 9939

Filed: April 07, 2000

For: CROSS-PLATFORM SUBSELECT METADATA EXTRACTION

Docket No: A8496 / ST9-99-179

Group Art Unit: 2172

Examiner: Hung Q. Pham

APPEAL BRIEF UNDER 37 C.F.R. § 41.37

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

10/24/2005 SZEWDIE1 00000025 09545592
01 FC:1402 500.00 OP

Sir:

In accordance with the provisions of 37 C.F.R. § 41.37, Appellant submits the following:

Table of Contents

I. REAL PARTY IN INTEREST

The real party in interest is INTERNATIONAL BUSINESS MACHINES CORPORATION by virtue of an assignment executed by Richard Henry Mandel III (hereinafter "Appellant") on April 4, 2000.

II. RELATED APPEALS AND INTERFERENCES

To the best of the knowledge and belief of the Appellant, the Assignee and the undersigned, there are no other appeals or interferences before the Board of Appeals and Interferences ("the Board") that will directly affect, or be affected by, the Board's decision in the present Appeal.

III. STATUS OF CLAIMS

Claims 1-39 are all the claims pending in the Application.

Claims 1-5, 8-16, 19-27, 30-33, 34, 36 and 38 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over what the Examiner has alleged to be Appellant's "*Admitted Prior Art*" (hereinafter "*Related Art*") set forth in the "related art" section of the Application (pgs. 1-2) in view of "*Fundamentals of Database Systems*" by *Elmasri et al.* (hereafter "*Elmasri*").

Claims 6, 7, 17, 18, 28, 29, 35, 37 and 39 stand objected to as being dependent upon rejected claims, but are indicated as being allowable if rewritten in independent form.

IV. STATUS OF AMENDMENTS

A *Response Under 37 C.F.R. § 1.116* was filed on July 27, 2005, in response to the Final *Office Action* dated April 25, 2005. The July 27, 2005 *Response* made no changes to the claims. No other amendment or response was filed subsequent to the April 25, 2005 Final *Office Action*.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

To explain the invention for the Board's convenience, Appellants will first describe the relevant art (pp. 1-2 of the Specification), and then the exemplary embodiments of the invention (pp. 3-21 of the Specification). Portions of the claims that correspond to the features shown in the exemplary embodiments are also referenced during this discussion (portions of independent claims 1, 12, 23, 34, 36 and 38 are provided in block quotes for easy identification). This discussion of the exemplary embodiments and the pending claims is provided for explanatory purposes only, and is not intended to limit the scope of the claims.

V(I). Relevant Art

The instant Application relates in general to a technique for cross-platform metadata (*i.e.*, information about other information) extraction (p. 1, lines 4-5).

A Relational Database Management System (RDBMS) is a database management system (DBMS) which uses relational techniques for storing and retrieving data. Relational databases are organized into tables, which consist of rows (*i.e.*, tuples or records) and columns of data. It is useful to obtain column names and data type of the tables (p. 1, lines 8-19).

Some related art systems exist for obtaining such column names and data types (*i.e.*, metadata), such as the DB2 system's DESCRIBE command in UNIX, Windows NT, and OS/2 versions. However, some systems do not have such functionality, such as the DB2 OS/390® version. Nevertheless, in such a system, it may be possible for a developer to alter a Data Manipulation Language (DML) statement so that it returns no data, but allows full access to the metadata similar to that provided in the above DESCRIBE command (p. 1, line 20 - p. 2, line 2).

Appellants have identified a need in the art of an improved technique of obtaining this information, which is available across platforms (p. 2, lines 14-15).

V(II). Exemplary Embodiments of the Invention

Accordingly, Appellant's invention is directed to a system, as shown in FIG. 1, that includes client computers 102 (*e.g.*, a PC or workstation), server computers 104 (*e.g.*, a PC, workstation, minicomputer, or mainframe), data sources 106, and a bi-directional network 100 (*e.g.*, a LAN, WAN or internet) (p. 3, line 22 - p. 5, line 5).

FIG. 2 (reproduced to the right) illustrates a client computer 200 that operates under the control of an operating system (OS) 214 in memory 204 of client computer 200. The operating system 214 controls the execution of one or more computer programs 216, such as a Stored Procedure Builder 218, which includes a Metadata Extraction System 220 (p. 5, line 6 - p. 6, line 17).

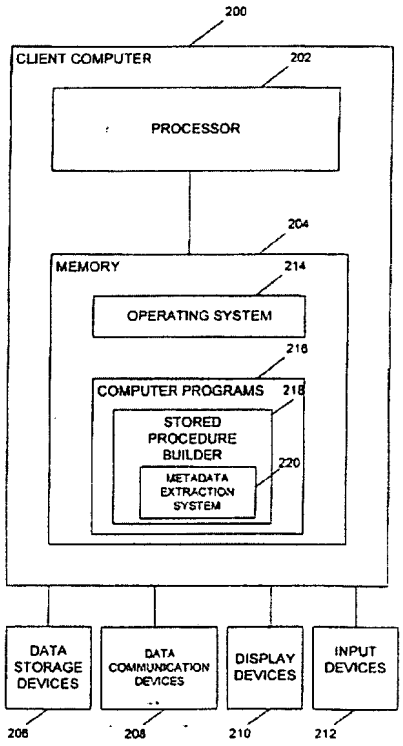

FIG. 2

FIG. 3 (reproduced to the right) illustrates a server including computer system 302 and data storage devices 304 and 306 that store relational databases. Computer system 302 includes Internal Resource Lock Manager (IRLM) 310 (for handling locking services), the Systems Services module 312 (for environment control), and the

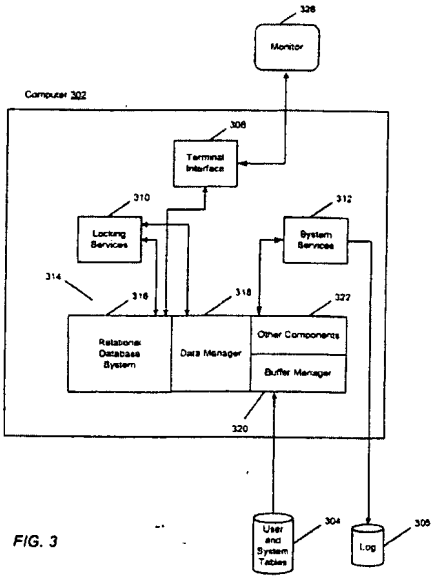

FIG. 3

Database Services module 314. Database Services module 314 contains several submodules, including the Relational Database System (RDS) 316, the Data Manager 318, the Buffer Manager 320, and other components 322 such as a SQL compiler/interpreter. These submodules support the functions of the SQL language (*i.e.*, definition, access control, interpretation, compilation, database retrieval, and update of user and system data) (p. 6, line 18 - p. 8, line 16).

The above corresponds to the following portions of independent claims 1, 12 and 23:

1. A method for executing a query against a database on a data storage device connected to a computer, the method comprising ...	12. An apparatus for executing a query, comprising: a computer connected a data storage device that stores a database containing data ...	23. An article of manufacture comprising a computer program carrier readable by computers and embodying one or more instructions executable by the computer for executing a query against a database on a data storage device connected to the computer, comprising ...

The instant Application is generally directed to using SQL statements executed under the control of the Database Services module 314. The utilized SQL statement may be an SQL SELECT statement containing certain clauses (*e.g.*, WHERE and GROUPBY clauses) (p. 8, lines 19-25; p. 7, lines 12-17).[1, 2, 3]

The metadata extraction system 220 extracts metadata across platforms (*i.e.*, without knowledge of which platform the Metadata Extraction System 220 is running on) by modifying the SQL statement to replace those WHERE and/or GROUPBY clauses with a logically false statement (p. 10, line 26 - p. 11, line 15).

---

[1] This supports claim 2, 13 and 24's recitation that "the query comprises a SELECT statement."

[2] This supports claim 4, 15 and 26's recitation that "the selected clauses comprise WHERE clauses."

[3] This supports claim 5, 16 and 27's recitation that "the selected clauses comprise GROUP BY clauses."

As a matter of example, the SELECT statement may be first modified by the metadata extraction system 220 to include a logically false statement such as WHERE 1 = -1. In this example 1 can never equal –1 and therefore the WHERE clause is evaluated to be a false condition (p. 10, lines 26-27).

The above corresponds to the following portions of independent claims 1, 12 and 23:

1. ... modifying the query to replace one or more selected clauses with one or more false clauses ...	12. ... one or more computer programs, performed by the computer, for modifying the query to replace one or more selected clauses with one or more false clauses ...	23. ... modifying the query to replace one or more selected clauses with one or more false clauses ...

Then, the modified SQL clause, including the logically false statement such as WHERE 1 = -1, is executed, and no data is returned. This is because no row in the result set can satisfy the logically false condition. However, metadata (the column type) for any data that would be returned if the false condition were not present, is returned (p. 11, lines 1-4).[4]

The above corresponds to the following portions of independent claims 1, 12 and 23:

1. ... executing the modified query with the one or more false clauses; and retrieving metadata from the result set obtained by executing the modified query.	12. ... one or more computer programs, performed by the computer, for ... executing the modified query with the one or more false clauses, and retrieving metadata from the result set obtained by executing the modified query.	23. ... executing the modified query with the one or more false clauses; and retrieving metadata from the result set obtained by executing the modified query.

In one embodiment, the number of false SQL statements will be equivalent to the number of WHERE clauses. For example, if a SQL statement has a WHERE clause, then one false WHERE clause is generated and used to replace the original WHERE clause and anything following it (e.g., a GROUP BY clause). If a SQL statement has two WHERE clauses, then one

[4] This supports claim 8, 19 and 30's recitation that "the metadata comprises column type data for the result set."

false WHERE clause is generated and used to replace both of the WHERE clauses and everything following them and a second false WHERE clause is generated and used to replace the second WHERE clause and everything following it (i.e., this leaves the first WHERE clause in the SQL statement. The first SQL statement with a false WHERE clause to run without experiencing a SQL exception is the one used to extract the MetaData (p. 11, lines 11-15 and 21-24).[5, 6]

Additionally, the Metadata Extraction System 220 improves performance by removing the ORDER BY clauses, should they exist in the statement (p. 11, lines 16-20).

Accordingly, by modifying a query statement, such as an SQL statement, to include a false clause and executing the modified query, metadata can be returned even in computing platforms that do not have a metadata extraction command (p. 10, lines 21-25).

---

[5] This supports claim 6, 17 and 28's recitation that "modifying the query comprises: generating a list of modified queries, wherein each modified query has one or more selected clauses replaced with one or more false clauses; and executing each modified query until one executes successfully.

[6] This supports claim 7, 18 and 29's recitation that "a query executes successfully if it executes without an exception."

As a matter of further explanation, FIG. 6 (reproduced to the right) is a flow diagram of the processing performed by the Metadata Extraction System 220. First, the Metadata Extraction System 220 receives a SQL statement (600). Then, the Metadata Extraction System 220 processes WHERE and/or GROUP BY clauses in the SQL statement to create a list of SQL statements with false clauses (602). Next, the Metadata Extraction System 220 executes each statement in list in sequence until one executes successfully (604). Next, the Metadata Extraction System 220 obtains type data for columns from the result set of the successfully executed statement (606). Then, the Metadata Extraction System 220 converts the type data into JAVA types (608).[7] Lastly, the Metadata Extraction System 220 creates a SQLJ iterator with parameters having the JAVA types (610) [8, 9] (p. 13, line 21 - p. 14, line 2).[10]

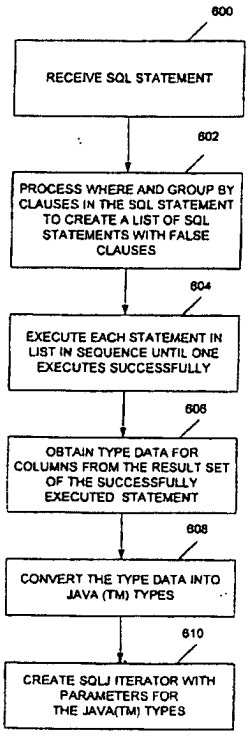

FIG. 6

[7] This supports claim 9, 20 and 31's recitation of "converting the column type data to JAVA types."

[8] This supports claim 10, 21 and 32's recitation of "generating a SQLJ iterator with parameters having the JAVA types."

[9] This supports claim 11, 22 and 33's recitation of "determining whether the query requires a SQLJ iterator."

[10] A SQL statement requires an iterator if it is a SELECT statement, other than a SELECT INTO statement (which selects data into one or more variables). This supports claim 3, 14 and 25's recitation that "the SELECT statement is not a SELECT INTO statement."

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether or not claims 1-5, 8-16, 19-27, 30-33, 34, 36 and 38 are patentable over the *Related Art* in view of *Elmasri*.

VII. ARGUMENT

VII(1). The Current Rejection

As noted above, the Examiner has rejected claims 1-5, 8-16, 19-27, 30-33, 34, 36 and 38 under 35 U.S.C. § 103(a) as being unpatentable over the instant Application's *Related Art* in view of *Elmasri*. Appellant disagrees with this rejection, and respectfully traverses it as follows.

VII(2) The Examiner's Reading of the Instant Application's Related Art Is Incorrect

Regarding independent claims 1, 12, 23, 35, 27 and 39, the Examiner first alleges, *inter alia*, that (*Office Action*, par. bridging pages 5 and 6):

> [the related art] teaches some databases have DESCRIBE command to list column names and data types of a query result. If the system does not have the DESCRIBE command, a developer can alter the DML statement to return no data but allow full access to the metadata to obtain the type of a result set.

Appellant respectfully disagrees with the Examiner's interpretation of the "Related Art" section of the instant Application. Specifically, the Examiner seems to read the statement (see pg. 2 of the specification) that "a developer can alter the Data Manipulation Language (DML) statement so that it returns no data," (see pg. 2 of the specification) as an admission that it would have been obvious to make such an alteration. Such a reading is contrary to the specific language of the statement.

Appellant has made no such admission that anyone has ever altered a DML statement in the manner described. Rather, the statement in question is merely a statement of capability, as Appellant specifically used the term "can" and not "have." Thus, the "Related Art" section of the specification does not teach or suggest that any such modification of a DML statement has been performed, or that anyone prior to the inventor of the instant Application ever thought to

make such a modification. Accordingly, this statement cannot provide the requisite motivation to support that Examiner's rejection, and no other prior art reference is cited by the Examiner to provide such motivation.

Additionally, the Examiner's assertion that one "must" modify such a DML statement in systems that do not have a DESCRIBE function is misplaced and unsupported. It is respectfully submitted that other methods, could be used to determine metadata.

VII(3) The Applied References Fail to Teach or Suggest the Use of a False Clause

In further regard to independent claims 1, 12, 23, 35, 27 and 39, the Examiner alleges, *inter alia*, that (*Office Action*, par. bridging pages 5 and 6):

> As shown in FIG. 7.2(a) of page 194 [of *Elmasri*] is [sic] the result of query Q0. The result of the query includes column names as metadata and data because the condition of [sic] WHERE statement is TRUE. As shown in FIG. 7.3(c) is [sic] the result of query Q12 with column names as metadata but without any data return, because the condition of the WHERE statement is FALSE. Thus, the DML statement as disclosed in the admission, obviously, could be altered by replacing a WHERE statement with one or more false statements so that it returns no data

Appellant respectfully disagrees with the entire premise that: (1) query Q0 somehow has a TRUE clause therein while query Q12 somehow has a FALSE clause therein; and (2) that the result sets of Tables 7.2(a) and 7.3(c) can somehow show that the queries have such TRUE or FALSE clauses, as the Examiner alleges.

First, neither query is disclosed in *Elmasri* as having any clause that is, itself, true or false. For example, the WHERE clause of Query Q12 in *Elmasri* is:

> WHERE ESSN=E.SSN AND E.FNAME=DEPENDENT_NAME AND SEX=E.SEX).

No "false clause" is provided in this WHERE clause. Rather, each of the WHERE clauses in this string is set to a particular value, and is not a "false" clause, as there is nothing

"false" (or, for that matter, "true") about ESSN=E.SSN, E.FNAME=DEPENDENT_NAME, or SEX=E.SEX.

Second, while Appellant agrees that Table 7.3(c) shows the result of Query Q12 as applied against the tables of Figure 6.6, and that this illustrated result is a null set (*i.e.*, no values are returned), Appellant disagrees that this null set result is "because the condition of the WHERE statement is FALSE" as the Examiner alleges. Rather, in this case, the null set is returned in Table 7.3(c) because no records in the tables of Figure 6.6 match the parameters of Query Q12. However, the result set would be different if one of the records in the tables of Figure 6.6 did match the parameters of Query Q12, as it would then be displayed in Table 7.3(c). Thus, the result set obtained by Query Q12 is source table dependent as to whether the result set displays records or not.

In contrast, the presence of a "false" clause in query Q12 would, by definition, prevent any records from ever being displayed in the result table. In such a case, it would be completely irrelevant what records are contained in the table that the query is applied against, as no records would ever be returned. Thus, the result set obtained by the use of a "false" clause is source table independent, in contrast to the result set obtained by Query Q12.

Accordingly, even though the results of: (1) the claimed query with a false clause therein; and (2) a query that returns no matching records, such as Query Q12 of *Elamsri*, may appear similar, the ways these results are obtained, which is what is claimed in the instant Application, are very different. In other words, Appellants respectfully submit that, even though the result sets of Table 7.3(c) and of the method or system of the independent claims may be similar, none of the applied references teach or suggest actually replacing a selected clause with a false clause,

as recited in independent claims 1, 12, 23, 34, 36 and 38 of the instant Application, to obtain such a result set.

Thus, Appellant respectfully submits that independent claims 1, 12, 23, 34, 36 and 38 are patentable over the *Related Art*.

VII.(4) The Applied References Fail to Teach or Suggest the Features Recited in the Dependent Claims

Appellant respectfully submits that rejected dependent claims 2-5, 8-11, 13-16, 19-22, 24-27 and 30-33 are: (1) allowable at least by virtue of their dependency; and (2) separately patentable over the applied references.

For example, Appellant respectfully submits that the applied references fail to teach or suggest claim 5's recitation that "the selected clauses comprise GROUP BY clauses." While the Examiner alleges that the recited "GROUP BY clause" is disclosed in the *Related Art* or *Elmasri*, no portion of the *Related Art* or *Elmasri* even mentions a GROUP BY clause.

Appellant respectfully submits that claims 16 and 27, which recite features similar to claim 5, are separately patentable for the same reasons.

Further, Appellant respectfully submits that the applied references fail to teach or suggest claim 9's recitation of "converting the column type data to JAVA types." While the Examiner alleges that the recited "converting" is disclosed in the *Related Art*, no portion of the *Related Art* even mentions converting column type data into any format, let alone a JAVA type.

Appellant respectfully submits that claims 20 and 31, which recite features similar to claim 9, are separately patentable for the same reasons.

Further, Appellant respectfully submits that the applied references fail to teach or suggest claim 10's recitation of "generating a SQLJ iterator with parameters having the JAVA types."

While the Examiner alleges that the recited "generating" of an "SQLJ iterator" is disclosed in the *Related Art*, no portion of the *Related Art* even mentions "generating" any particular feature, let alone an SQLJ iterator.

Appellant respectfully submits that claims 21 and 32, which recite features similar to claim 10, are separately patentable for the same reasons.

Still further, Appellant respectfully submits that the applied references fail to teach or suggest claim 11's recitation of "determining whether the query requires a SQLJ iterator." While the Examiner alleges that the recited "determining" is disclosed in the *Related Art*, no portion of the *Related Art* even mentions "determining" any particular feature, let alone whether the query requires an SQLJ iterator.

Appellant respectfully submits that claims 22 and 33, which recite features similar to claim 11, are separately patentable for the same reasons.

VIII. CONCLUSION

In view of the foregoing differences between appealed claims 1-5, 8-16, 19-27, 30-33, 34, 36 and 38 and the applied references, Appellants respectfully submit that appealed claims 1-5, 8-16, 19-27, 30-33, 34, 36 and 38 are patentable over the applied references. Accordingly, Appellants respectfully submit that all of the pending claims 1-39 of the instant Application are patentable over the applied references.

Unless a check is submitted herewith for the fee required under 37 C.F.R. §41.37(a) and 1.17(c), please charge said fee to Deposit Account No. 19-4880.

The USPTO is directed and authorized to charge all required fees, except for the Issue Fee and the Publication Fee, to Deposit Account No. 19-4880. Please also credit any overpayments to said Deposit Account.

Respectfully submitted,

#47,177

Timothy P. Cremen
Registration No. 50,855

SUGHRUE MION, PLLC
Telephone: (202) 293-7060
Facsimile: (202) 293-7860

WASHINGTON OFFICE
23373
CUSTOMER NUMBER

Date: October 21, 2005

CLAIMS APPENDIX

CLAIMS 1-42 ON APPEAL:

1. (Previously Presented) A method for executing a query against a database on a data storage device connected to a computer, the method comprising:

modifying the query to replace one or more selected clauses with one or more false clauses;

executing the modified query with the one or more false clauses; and

retrieving metadata from the result set obtained by executing the modified query.

2. (Original) The method of claim 1, wherein the query comprises a SELECT statement.

3. (Original) The method of claim 2, wherein the SELECT statement is not a SELECT INTO statement.

4. (Original) The method of claim 1, wherein the selected clauses comprise WHERE clauses.

5. (Original) The method of claim 1, wherein the selected clauses comprise GROUP BY clauses.

6. (Previously Presented) The method of claim 1, wherein modifying the query comprises:

generating a list of modified queries, wherein each modified query has one or more selected clauses replaced with one or more false clauses; and

executing each modified query until one executes successfully.

7. (Original) The method of claim 6, wherein a query executes successfully if it executes without an exception.

8. (Original) The method of claim 1, wherein the metadata comprises column type data for the result set.

9. (Original) The method of claim 8, further comprising converting the column type data to JAVA types.

10. (Original) The method of claim 9, further comprising generating a SQLJ iterator with parameters having the JAVA types.

11. (Original) The method of claim 1, further comprising determining whether the query requires a SQLJ iterator.

12. (Previously Presented) An apparatus for executing a query, comprising:

a computer connected a data storage device that stores a database containing data;

one or more computer programs, performed by the computer, for modifying the query to replace one or more selected clauses with one or more false clauses, executing the modified query with the one or more false clauses, and retrieving metadata from the result set obtained by executing the modified query.

13. (Original) The apparatus of claim 12, wherein the query comprises a SELECT statement.

14. (Original) The apparatus of claim 13, wherein the SELECT statement is not a SELECT INTO statement.

15. (Original) The apparatus of claim 12, wherein the selected clauses comprise WHERE clauses.

16. (Original) The apparatus of claim 12, wherein the selected clauses comprise GROUP BY clauses.

17. (Previously Presented) The apparatus of claim 12, wherein modifying the query comprises:

generating a list of modified queries, wherein each modified query has one or more selected clauses replaced with one or more false clauses; and

executing each modified query until one executes successfully.

18. (Original) The apparatus of claim 17, wherein a query executes successfully if it executes without an exception.

19. (Original) The apparatus of claim 12, wherein the metadata comprises column type data for the result set.

20. (Original) The apparatus of claim 19, further comprising converting the column type data to JAVA types.

21. (Original) The apparatus of claim 20, further comprising generating a SQLJ iterator with parameters having the JAVA types.

22. (Original) The apparatus of claim 12, further comprising determining whether the query requires a SQLJ iterator.

23. (Previously Presented) An article of manufacture comprising a computer program carrier readable by computers and embodying one or more instructions executable by the computer for executing a query against a database on a data storage device connected to the computer, comprising:

modifying the query to replace one or more selected clauses with one or more false clauses;

executing the modified query with the one or more false clauses; and

retrieving metadata from the result set obtained by executing the modified query.

24. (Original) The article of manufacture of claim 23, wherein the query comprises a SELECT statement.

25. (Original) The article of manufacture of claim 24, wherein the SELECT statement is not a SELECT INTO statement.

26. (Original) The article of manufacture of claim 23, wherein the selected clauses comprise WHERE clauses.

27. (Original) The article of manufacture of claim 23, wherein the selected clauses comprise GROUP BY clauses.

28. (Previously Presented) The article of manufacture of claim 23, wherein modifying the query comprises:

generating a list of modified queries, wherein each modified query has one or more selected clauses replaced with one or more false clauses; and

executing each modified query until one executes successfully.

29. (Original) The article of manufacture of claim 28, wherein a query executes successfully if it executes without an exception.

30. (Original) The article of manufacture of claim 23, wherein the metadata comprises column type data for the result set.

31. (Previously Presented) The article of manufacture of claim 30, further comprising converting the column type data to JAVA types.

32. (Previously Presented) The article of manufacture of claim 31, further comprising generating a SQLJ iterator with parameters having the JAVA types.

33. (Original) The article of manufacture of claim 23, further comprising determining whether the query requires a SQLJ iterator.

34. (Previously Presented) A method for executing a query against a database on a data storage device connected to a computer, the method comprising:

modifying the query to replace one or more selected clauses with a false clause;

executing the modified query with the false clause; and

retrieving metadata from the result set obtained by executing the modified query.

35. (Previously Presented) The method of claim 34, wherein modifying the query comprises:

generating a list of modified queries, wherein each modified query has one or more selected clauses replaced with a false clause; and

executing each modified query until one executes successfully.

36. (Previously Presented) An apparatus for executing a query, comprising:

a computer connected a data storage device that stores a database containing data;

one or more computer programs, performed by the computer, for modifying the query to replace one or more selected clauses with a false clause, executing the modified query with the false clause, and retrieving metadata from the result set obtained by executing the modified query.

37. (Previously Presented) The apparatus of claim 36, wherein modifying the query comprises:

generating a list of modified queries, wherein each modified query has one or more selected clauses replaced with a false clause; and

executing each modified query until one executes successfully.

38. (Previously Presented) An article of manufacture comprising a computer program carrier readable by computers and embodying one or more instructions executable by the

computer for executing a query against a database on a data storage device connected to the computer, comprising:

modifying the query to replace one or more selected clauses with a false clause;

executing the modified query with the false clause; and

retrieving metadata from the result set obtained by executing the modified query.

39. (Previously Presented) The article of manufacture of claim 38, wherein modifying the query comprises:

generating a list of modified queries, wherein each modified query has one or more selected clauses replaced with a false clause; and

executing each modified query until one executes successfully.

EVIDENCE APPENDIX

This Appendix is Not Applicable to the instant Appeal.

RELATED PROCEEDINGS APPENDIX

This Appendix is Not Applicable to the instant Appeal.